FINDING PATHS AND CYCLES IN GRAPHS

SERGEY GUBIN

Abstract. A polynomial time algorithm which detects all paths and cycles of all lengths in form of vertex pairs (start, finish).

Introduction

Problem of path/cycle existence and finding is a big topic in Discrete Mathematics and Computer Science [1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 15, 16, 17, and others]. This article is not a review but just another design.

The design's idea may be traced back to Sir William R. Hamilton, [1]: "I have lately been led to the conception of a new system, or rather family of systems, of non-commutative roots of unity, ... which admit, even more easily than the quaternion symbols do, of geometrical interpretation. ... every one of the results may be interpreted, as having reference to the passage from face to face (or from corner to corner) of the icosahedron (or of the dodecahedron) ... ." Let us rephrase: a formal system was what allowed tracking of the "passages".

This work models a graph's walks with a generative grammar and applies dynamic programming to detect paths/cycles. Some deviation from the widely accepted terminology is due to the grammar: walks are just sequences of adjacent arcs, arcs/loops are walks/paths/circuits/cycles of length 1, etc. Author hopes that this will not be a nuisance. Also, any graph becomes a multi graph when talking walks. So, let us start with that from the very beginning.

1. Walk generator

Let $g = (V, A)$ be a given multi digraph (possible with loops): V is the vertex set and A is the arc set of the graph. Let's arbitrarily enumerate vertices:

$$V = \{v_1, v_2, \dots, v_n\},$$

- and build a set matrix $S = (s_{ij})_{n\times n}$ whose (i, j)-element is the set of all arcs which start in vertex v_i and finish in vertex v_j; some of the elements can be $\emptyset$.

Let's define powers of matrix S:

$$S^1 = S,\ S^{k+1} = S^k S,\ k \geq 1.$$

The multiplication of set matrices is performed by the rules of the usual matrix multiplication but with replacement of the numeric products with Cartesian products of sets and the numerical sum with set joins: if $X = (x_{ij})_{a\times b}$ and $Y = (y_{ij})_{b\times c}$

Date: August 30, 2007.
2000 *Mathematics Subject Classification.* Primary 05C38, Secondary 68R10.
Key words and phrases. graph, path, cycle, Hamiltonian, clique, NP-complete.

are set matrices, then

$$XY = (\bigcup_{\beta=1}^{b} x_{i\beta} \times y_{\beta j})_{a\times c}.$$

Let us mention that the set matrix multiplication is an associative operation. For matrix S^k, the last formula gives:

$$S^1 = S,\ S^{k+1} = (\bigcup_{\beta=1}^{n} (S^k)_{i\beta} \times (S)_{\beta j})_{n\times n},\ k \geq 1. \tag{1.1}$$

Here and further, symbol $(M)_{ij}$ means (i,j)-element of matrix M.

Matrix S is a walk generator: element $(S^k)_{ij}$ consists of all such walks "written" in arcs, which have length k, start in vertex v_i, and finish in vertex v_j; all circuits of length k are assembled in diagonal elements. That may be cleared with the following decomposition:

$$(S^{k+1})_{ij} = \bigcup_{\beta=1}^{n} (S^k)_{i\beta} \times (S)_{\beta j} =$$

$$\bigcup_{\beta=1}^{n} (\bigcup_{\gamma=1}^{n} (S^{k-1})_{i\gamma} \times (S)_{\gamma\beta}) \times (S)_{\beta j} = \bigcup_{\beta,\gamma=1}^{n} (S^{k-1})_{i\gamma} \times (S)_{\gamma\beta} \times (S)_{\beta j} = \ldots =$$

$$= \bigcup_{\sigma} (S)_{\sigma_1=i,\sigma_2} \times (S)_{\sigma_2\sigma_3} \times \ldots (S)_{\sigma_{k-1}\sigma_k} \times (S)_{\sigma_k,\sigma_{k+1}=j}, \tag{1.2}$$

- where join is taken over all n^{k-1} samples $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_{k+1})$ of $k+1$ numbers constrained with the following:

$$\sigma_1 = i;\ 1 \leq \sigma_m \leq n,\ m = 2,3,...,k;\ \sigma_{k+1} = j. \tag{1.3}$$

Let $(S_P^{k+1})_{ij}$ be a limitation of join 1.2 to those samples σ which are permutations; and $(S_C^{k+1})_{ii}$ be a limitation of join 1.2 to those samples σ which are circular permutations.

Theorem 1.1. *In multi digraph g:*
1. There is a walk of length $k > 1$ from v_i into v_j iff $(S^k)_{ij} \neq \emptyset$;
2. There is a path of length $k > 1$ from v_i into v_j iff $(S_P^k)_{ij} \neq \emptyset$;
3. There is a cycle of length $k > 1$ attached to v_i iff $(S_C^k)_{ii} \neq \emptyset$;
4. There are paths and cycles of length 1 iff appropriate elements of walk generator $S = S^1$ are not empty.

Exercise 1.2. Suppose g has two vertices $v_1,\ v_2$ and four arcs $a_{11},\ a_{12},\ a_{21}$, and a_{22}: arc a_{ij} begins in vertex v_i and ends in vertex v_j. Then, the walk generator for g is the following set matrix:

$$S^1 = S = \begin{pmatrix} \{a_{11}\} & \{a_{12}\} \\ \{a_{21}\} & \{a_{22}\} \end{pmatrix}.$$

The arcs/loops are walks of length 1. Arcs a_{12} and a_{21} are paths of length 1. Loops a_{11} and a_{22} are cycles of length 1.

Walks of length 2 constitute elements of set matrix S^2:

$$S^2 = SS = \begin{pmatrix} \{a_{11}a_{11}, a_{12}a_{21}\} & \{a_{11}a_{12}, a_{12}a_{22}\} \\ \{a_{21}a_{11}, a_{22}a_{21}\} & \{a_{21}a_{12}, a_{22}a_{22}\} \end{pmatrix}.$$

There are no paths of length 2. Diagonal elements are circuits of length 2. Among them is cycle $(a_{12}a_{21}) = (a_{21}a_{12})$. It is a Hamiltonian cycle.

Walks of length 3 constitute elements of set matrix $S^3 = S^2S$, and so on.

Iterations 1.1 create a language - we may call it an arc language. Its alphabet is A, and its grammar is a context-sensitive L-system whose axiom is walk generator S and whose production rules, in essence, consist of omitting the curly braces in formula 1.1 and replacing $\emptyset$ with empty string. The k-th generation of the system consists of all walks/strings of length k written in arcs. The walks are sorted over vertex pairs (start, finish).

The exact numbers of k-walks in these vertex pairs (start, finish) are elements of k-th power of the multi digraph's adjacency matrix. From formula 1.2, a rough upper bound for the numbers is

$$|V|^{k-1}(\max_{ij}|S_{ij}|)^k.$$

The bound is reachable in complete graphs. General walk pattern is a path with cycles attached, and multiplicand $|V|^{k-1}$ is due to the branching. Nevertheless, using formula 1.1 can be an efficient method for particular problems/graphs - sparse digraphs, trees, and others with simple walk pattern.

Exercise 1.3. In the example from exercise 1.2, there are 2^{k+1} walks of length k, $k \geq 1$, because there is cycle $a_{12}a_{21}$ with two cycles/loops attached to it - a_{11} and a_{22}. Written in arcs, the pattern of walks from v_i into v_j is a self-similar structure described by the following grammar:

$$[\ [a_{ii}]a_{ij}[a_{jj}]a_{ji}\ ]\ a_{ij}\ [\ [a_{jj}]a_{ji}[a_{ii}]a_{ij}\ ],$$

- where symbol $[x]$ means zero or more repetitions of x.

For the path/cycle problem, the algorithmic simplicity of formula 1.1 looks tempting: all paths/cycles could be found with filtering of all n^2 sets $(S^k)_{ij}$. Although those sets have exponential power, filtering per se could be made efficient due to a possibility of sorting/filtering the sets during iterations 1.1. That which makes the method inefficient is a need for calculation/remembering of exponential size sets - elements of S^k. The problem can be tried and fought with walk coloring.

2. Walk coloring

Walk colors are a semantics of the arc language defined by formula 1.1. Coloring can be done with an arc coloring; a function defined on walks; and with other means, as well. For example: adjacency matrix is a coloring with a "number of walks" function; transitional diagram is a digraph arc-colored with the events triggering transitions; Markov chain is a digraph arc-colored with the transitional probabilities; the Icosian Calculus [1, 2] is the icosahedron colored with the Hamilton's symbols, etc. The original walk generator defined in section 1 is a coloring appropriate to the identity function on set of all walks. Its opposite is a monochromatic coloring - all walks have the same color.

Exercise 2.1. In the example from exercise 1.2, let's color the arcs starting in v_1 with color "Head" and the arcs starting in v_2 with color "Tail". Also, let's use the string presentation of iterations 1.1 described in that section. Then, the example

will become an algorithmic model of coin flipping:

$$S = \begin{pmatrix} H & H \\ T & T \end{pmatrix},\ S^2 = \begin{pmatrix} HH, HT & HH, HT \\ TH, TT & TH, TT \end{pmatrix},\ \cdots$$

Changing of the colors to "0" and "1" will make the example a generator of all/random finite $(0,1)$-sequences.

Exercise 2.2. Let P be the following monochromatic arc coloring of g:

$$(P)_{ij} = \begin{cases} 1, & (S)_{ij} \neq \emptyset \\ 0, & (S)_{ij} = \emptyset \end{cases}.$$

Then, number of different Hamiltonian cycles in g is

$$\frac{1}{n}\sum_{\sigma}(P)_{\sigma_1\sigma_2}(P)_{\sigma_2\sigma_3}\dots(P)_{\sigma_{n-1}\sigma_n}(P)_{\sigma_n\sigma_1},$$

where sum is taken over all circular permutations $\sigma = (\sigma_1, \sigma_2, \dots, \sigma_n)$ of numbers $1, 2, \dots, n$.

In the monochromatic case, color is just a sign of adjacency. The coloring can be modeled with Boolean matrix - a matrix filled with $true$ and $false$ (for $\emptyset$). Then, iterations 1.1 can be replaced with formulas for powers of that Boolean matrix. The powers are calculated with the algorithm for numeric matrix multiplication except the products are replaced with conjunctions and the sums are replaced with disjunctions: if $X = (x_{ij})_{a\times b}$ and $Y = (y_{ij})_{b\times c}$ are Boolean matrices, then

$$XY = (\bigvee_{\beta=1}^{b} x_{i\beta} \wedge y_{\beta j})_{a\times c}. \tag{2.1}$$

Let us mention that the Boolean matrix multiplication is an associative operation.

Boolean arc coloring is as follows:

$$(B^1)_{ij} = \begin{cases} true, & (S)_{ij} \neq \emptyset \\ false, & (S)_{ij} = \emptyset \end{cases} \tag{2.2}$$

And Boolean walk coloring is defined with the following iterations:

$$B^{k+1} = (\bigvee_{\beta=1}^{n} (B^k)_{i\beta} \wedge (B^1)_{\beta j})_{n\times n},\ k \geq 1. \tag{2.3}$$

The $true/false$ coloring is efficient for the reachability problems: formula 2.3 finds all pairs of connected vertices with $(2n-1)n^3$ operations "$\vee$" and "$\wedge$" at most.

To prove that, let's perform decomposition of $(B^k)_{ij}$ exactly like for formula 1.2:

$$(B^{k+1})_{ij} = \bigvee_{\sigma}(B^1)_{\sigma_1=i,\sigma_2} \wedge (B^1)_{\sigma_2\sigma_3} \wedge \dots (B^1)_{\sigma_{k-1}\sigma_k} \wedge (B^1)_{\sigma_k,\sigma_{k+1}=j}, \tag{2.4}$$

- where disjunction is taken over all number samples σ which satisfy constrains 1.3.

Let $(B_P^{k+1})_{ij}$ be a limitation of disjunction 2.4 to those samples σ which are permutations; and $(B_C^{k+1})_{ii}$ be a limitation of disjunction 2.4 to those samples σ which are circular permutations. Then, theorem 1.1 implies

Theorem 2.3. *In multi digraph g:*
1. *There is a walk of length $k > 1$ from v_i into v_j iff $(B^k)_{ij} = true$;*
2. *There is a path of length $k > 1$ from v_i into v_j iff $(B_P^k)_{ij} = true$;*
3. *There is a cycle of length $k > 1$ attached to v_i iff $(B_C^k)_{ii} = true$;*

4. There are paths and cycles of length 1 *iff appropriate elements of Boolean matrix* B^1 *are true.*

Although Boolean coloring 2.2, 2.3 is inefficient for the path/cycle problem in multi digraph, it still can be efficient for other particular problems.

Exercise 2.4 (Shortest Path)**.** Length of the shortest path from vertex v_{i_0} into vertex v_{j_0} can be calculated with formula 2.3 by multiplication of i_0-th string of Boolean matrix B^k on whole Boolean matrix B^1 until element $(B^{k+1})_{i_0 j_0}$ will become *true*, $k = 1, 2, \ldots, n-2$. Then, the shortest path has length $k+1$. The table below compares the method with classical routing algorithms:

Algorithm	Formula 2.3	Bellman-Ford	Dijkstra
Time complexity	$O(\|V\|^3)$	$O(\|V\|\|A\|)$	$O(\|V\|^2 + \|A\|)$

Let us emphasize the algorithmic simplicity of formula 2.3 and the potential of that formula for further computational simplifications.

Exercise 2.5 (Maximum Clique)**.** Let Q be the following Boolean matrix:

$$(Q)_{ij} = \begin{cases} (B^1)_{ij} \wedge (B^1)_{ji}, & i > j \\ false, & i \leq j \end{cases},$$

- where B^1 is defined with formula 2.2. Let's define Q^m with the following iterations:

$$Q^1 = Q,\ Q^{m+1} = Q^m Q \wedge Q^m,\ m \geq 1$$

- where product of Boolean matrices is defined with formula 2.1, and conjunction of Boolean matrices is a matrix of conjunctions of appropriate elements. It may be seen that maximum clique in g has the following size:

$$q = \min_k \{k \mid Q^k = (false)_{n \times n}\}.$$

Let us clarify that any vertex is a clique of size 1 and sketch a proof:

Sketch of proof. Boolean matrix Q defines a subgraph of multi digraph g. This subgraph is a directed forest. All walks in this forest are paths. Boolean matrix Q^k rids that forest's arc set of all arcs except those vertex pairs (v_i, v_j) which are start and finish of paths of all length $1, 2, \ldots, k$. Cliques of size $k+1$ are the only invariants under such depletion of g. □

For multi digraph, the path/cycle problem's semantics is "to visit vertices only once". That gives an idea of efficient walk coloring for the problem. The idea is to make from the walk colors a memory. It is an interesting question, what to remember. Specter of options stretches from remembering visited vertices only to remembering unvisited vertices only. Disconnected graphs make the strategy question non-trivial. Nevertheless, those extreme strategies are dual and both based on an innate knowledge 1.1. Let's explore the rationalistic approach.

3. Path problem

Let's encode walks with the unvisited vertices. Two opposite strategies are possible: to remember arcs' start-vertices or to remember arcs' finish-vertices. The strategies are dual. They correspond to the ambiguity of words "visited vertex":

was it exited already or just entered?

Arc coloring/coding with start-vertices is as follows:

$$(F^1)_{ij} = \begin{cases} V - \{v_i\}, & (S)_{ij} \neq \emptyset \ \wedge \ i \neq j \\ \emptyset, & (S)_{ij} = \emptyset \ \vee \ i = j \end{cases}, \tag{3.1}$$

- where S the walk generator. Walk coloring with start-vertices is defined with the following iterations:

$$(F^{k+1})_{ij} = \begin{cases} \bigcup_{\beta=1}^{n} (F^k)_{i\beta} \cap (F^1)_{\beta j}, & i \neq j \\ \emptyset, & i = j \end{cases}, \ \ k \geq 1 \tag{3.2}$$

The formula may be seen as another set matrix multiplication.

Let's call that coloring defined with formulas 3.1 and 3.2 a F-coloring: all arcs which start from vertex v_i have the same color; the color is set $V - \{v_i\}$; the diagonal elements are rid of because they are loops/circuits; all walks from vertex v_i into vertex v_j have the same color; the color is set of all vertices which were missed by one or more walks from v_i into v_j - those vertices missed by a walk will extend that walk in the future when grammar 1.1 will allow that. To clarify that, let's perform decomposition of a non-empty set $(F^{k+1})_{ij}$:

$$(F^{k+1})_{ij} = \bigcup_{\beta=1}^{n} (F^k)_{i\beta} \cap (F^1)_{\beta j} =$$

$$= \bigcup_{\beta=1}^{n} \{ \bigcup_{\gamma=1}^{n} (F^{k-1})_{i\gamma} \cap (F^1)_{\gamma\beta} \} \cap (F^1)_{\beta j} = \bigcup_{\beta,\gamma=1}^{n} (F^{k-1})_{i\gamma} \cap (F^1)_{\gamma\beta} \cap (F^1)_{\beta j} = \ldots =$$

$$= \bigcup_{\sigma} (F^1)_{\sigma_1 = i, \sigma_2} \cap (F^1)_{\sigma_2 \sigma_3} \cap \ldots \cap (F^1)_{\sigma_{k-1}\sigma_k} \cap (F^1)_{\sigma_k, \sigma_{k+1} = j}, \tag{3.3}$$

- where join is taken over all number samples $\sigma = (\sigma_1, \sigma_2, \ldots, \sigma_{k+1})$ which satisfy constrains 1.3.

That which makes F-coloring special is the following inclusion:

$$(F^k)_{ij} \subseteq V. \tag{3.4}$$

It will be shown that this inclusion makes the coloring an efficient one.

Exercise 3.1. For the example from exercise 1.2:

$$F^1 = \begin{pmatrix} \emptyset & \{v_2\} \\ \{v_1\} & \emptyset \end{pmatrix},$$

$$F^2 = \begin{pmatrix} \emptyset & \emptyset \cap \{v_2\} \ \cup \ \{v_2\} \cap \emptyset \\ \{v_1\} \cap \emptyset \ \cup \ \emptyset \cap \{v_1\} & \emptyset \end{pmatrix} = \begin{pmatrix} \emptyset & \emptyset \\ \emptyset & \emptyset \end{pmatrix}.$$

Thus, due to formula 3.2, $F^k = (\emptyset)_{2\times 2}$ for all powers $k \geq 2$.

Lemma 3.2. *In multi digraph g, if there is a path of length $k \geq 1$ from vertex v_i into vertex v_j, then $(F^k)_{ij} \neq \emptyset$.*

Proof. Suppose, vertices

$$v_{\mu_1 = i}, v_{\mu_2}, \ldots, v_{\mu_k}, v_{\mu_{k+1} = j}$$

constitute a path of length k from v_i into v_j. Then,

$$v_{\mu_{k+1}} \neq v_{\mu_m}, \ m = 1, 2, \ldots, k.$$

Thus, by definition 3.1:

$$v_{\mu_{k+1}} \in (F^1)_{\mu_m \mu_{m+1}} \neq \emptyset,\ m = 1, 2, \ldots, k.$$

Then, due to decomposition 3.3:

$$v_{\mu_{k+1}} \in (F^1)_{\mu_1 \mu_2} \cap (F^1)_{\mu_2 \mu_3} \cap \ldots \cap (F^1)_{\mu_k \mu_{k+1}} \subseteq (F^k)_{\mu_1 \mu_{k+1}} = (F^k)_{ij} \neq \emptyset.$$

□

Lemma 3.3. *In multi digraph g, if $(F^k)_{ij} \neq \emptyset$, then there is a path of length k from vertex v_i into vertex v_j.*

Proof. Let's use mathematical induction over k.

For k = 1, by definition 3.1, inequality $(F^1)_{ij} \neq \emptyset$ implies

$$(S)_{ij} \neq \emptyset.$$

Then, there is an ark $a \in (S)_{ij}$. The arc is a path of length 1.

Let's assume that the lemma is true for $k = l$.

Let $k = l + 1$. Due to presentation 3.3, inequality $(F^k)_{ij} \neq \emptyset$ implies existence of such number sample $\sigma_0 = (\mu_2, \mu_3, \ldots, \mu_k)$ that

$$F_{\sigma_0} = (F^1)_{i=\mu_1, \mu_2} \cap (F^1)_{\mu_2 \mu_3} \cap (F^1)_{\mu_3 \mu_4} \cap \ldots \cap (F^1)_{\mu_{k-1}, \mu_k} \cap (F^1)_{\mu_k, \mu_{k+1}=j} \neq \emptyset.$$

Then, due to definition 3.1, there are k arcs

$$a_{\mu_m \mu_{m+1}} \in (S)_{\mu_m \mu_{m+1}},\ m = 1, 2, \ldots, k$$

- where S is the walk generator of g. The arcs create a walk w of length k from v_i into v_j:

$$w = (v_{\mu_1 = i}, a_{\mu_1 \mu_2}, v_{\mu_2}, a_{\mu_2 \mu_3}, \ldots, v_{\mu_k}, a_{\mu_k \mu_{k+1}} v_{\mu_{k+1}=j}).$$

Let's notice that

$$(F^{k-1})_{\mu_1 \mu_k} \supseteq (F^1)_{i=\mu_1, \mu_2} \cap (F^1)_{\mu_2 \mu_3} \cap (F^1)_{\mu_3 \mu_4} \cap \ldots \cap (F^1)_{\mu_{k-1}, \mu_k} \supseteq F_{\sigma_0} \neq \emptyset.$$

Thus, due to the induction hypothesis, the following sub-walk of walk w

$$(v_{\mu_1 = i}, a_{\mu_1 \mu_2}, v_{\mu_2}, a_{\mu_2 \mu_3}, \ldots, v_{\mu_k})$$

is a path of length $k - 1$ from $v_{i=\mu_1}$ into v_{μ_k}. From the other side,

$$(F^{k-1})_{\mu_2 \mu_{k+1}} \supseteq (F^1)_{\mu_2 \mu_3} \cap (F^1)_{\mu_3 \mu_4} \cap \ldots \cap (F^1)_{\mu_{k-1}, \mu_k} \cap (F^1)_{\mu_k, \mu_{k+1}=j} \supseteq F_{\sigma_0} \neq \emptyset,$$

as well. Thus, due to the induction hypothesis, the following sub-walk of walk w

$$(v_{\mu_2}, a_{\mu_2 \mu_3}, \ldots, v_{\mu_k}, a_{\mu_k \mu_{k+1}} v_{\mu_{k+1}=j})$$

is a path of length $k - 1$ from v_{μ_2} into $v_{\mu_{k+1}=j}$. Thus, if

$$v_i \neq v_j,$$

then walk w will be a path of length k from v_i into v_j. That last inequality follows from definition 3.2. □

Let us emphasize the role of grammar 1.1 on an example of a disconnected graph.

Exercise 3.4. Let's add to the digraph from exercise 1.2 vertex v_3, arc a_{23} from v_2 into v_3, and isolated vertex v_4. Walk generator for the resulting graph is

$$S = \begin{pmatrix} \{a_{11}\} & \{a_{12}\} & \emptyset & \emptyset \\ \{a_{21}\} & \{a_{22}\} & \{a_{23}\} & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix}.$$

There are three paths of length 1 in the digraph: $(v_1 a_{12} v_2)$, $(v_2 a_{21} v_1)$, and $(v_2 a_{23} v_3)$. Also, there is one path of length 2 in the digraph: $(v_1 a_{12} v_2 a_{23} v_3)$.

Arc F-coloring 3.1 detects the 1-paths:

$$F^1 = \begin{pmatrix} \emptyset & \{v_2, v_3, v_4\} & \emptyset & \emptyset \\ \{v_1, v_3, v_4\} & \emptyset & \{v_1, v_3, v_4\} & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix}.$$

Iterations 3.2 detect the 2-path and absence of the longer paths:

$$F^2 = \begin{pmatrix} \emptyset & \emptyset & \{v_3, v_4\} & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix};\ F^k = (\emptyset)_{4\times 4},\ k \geq 3.$$

Together, lemmas 3.2 and 3.3 constitute a theorem:

Theorem 3.5. *In multi digraph g, there is a path of length $k \geq 1$ from vertex v_i into vertex v_j iff $(F^k)_{ij} \neq \emptyset$.*

Let's estimate computational complexity of theorem 3.5. Due to inclusion 3.4, arc F-coloring 3.1 requires $O(n^3)$ operations. And walk F-coloring 3.2 requires $O(kn^4)$ operations: $O(n^2)$ operations at most to calculate each of n^2 elements $(F^m)_{ij},\ m = 1, 2, \ldots, k$. The total computational complexity of theorem 3.5 is

$$O(kn^4).$$

Particularly, when $k = n - 1$, theorem 3.5 solves the Hamilton path problem in time

$$O(n^5).$$

Obviously, the results are true when an "arcs' finish-vertices" coloring is used instead of the "arcs' start-vertices" coloring described. A finish-vertices coloring can be done with the following G-coloring. Arc G-coloring is defined as follows:

$$(G^1)_{ij} = \begin{cases} V - \{v_j\}, & (S)_{ij} \neq \emptyset \ \wedge \ i \neq j \\ \emptyset, & (S)_{ij} = \emptyset \ \vee \ i = j \end{cases}, \tag{3.5}$$

- where S is the walk generator. Walk G-coloring is defined with the following iterations

$$(G^{k+1})_{ij} = \begin{cases} \bigcup_{\beta=1}^{n} (G^k)_{i\beta} \cap (G^1)_{\beta j}, & i \neq j \\ \emptyset, & i = j \end{cases},\ k \geq 1 \tag{3.6}$$

4. Cycle problem

Let us define a H-coloring. Arc H-coloring is as follows:

$$(H^1)_{ij} = \begin{cases} (S)_{ij} & i = j \\ \emptyset, & i \neq j \end{cases}, \tag{4.1}$$

- where S is walk generator 1.1; and walk H-coloring is as follows:

$$(H^{k+1})_{ij} = \begin{cases} \bigcup_{\beta=1}^{n} (F^k)_{i\beta} \cap (G^1)_{\beta j}, & i = j \\ \emptyset, & i \neq j \end{cases}, \quad k \geq 1, \tag{4.2}$$

- where set matrices F and G were defined in section 3.

Exercise 4.1. For the graph from exercise 3.4:

$$G^1 = \begin{pmatrix} \emptyset & \{v_1, v_3, v_4\} & \emptyset & \emptyset \\ \{v_2, v_3, v_4\} & \emptyset & \{v_2, v_3, v_4\} & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix};$$

Then,

$$H^1 = \begin{pmatrix} \{a_{11}\} & \emptyset & \emptyset & \emptyset \\ \emptyset & \{a_{22}\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix};$$

$$H^2 = \begin{pmatrix} \{v_2, v_3, v_4\} & \emptyset & \emptyset & \emptyset \\ \emptyset & \{v_1, v_3, v_4\} & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \\ \emptyset & \emptyset & \emptyset & \emptyset \end{pmatrix}; \; H^k = (\emptyset)_{4\times 4}, \; k \geq 3.$$

Let's notice that non-empty elements of the coloring indicate all cycles in the digraph: loops a_{11}, a_{22} and 2-cycle $(v_1 a_{12} v_2 a_{21} v_1)$.

Lemma 4.2. *In multi digraph g, let p be a path of length k:*

$$p = (v_{\mu_1} a_{\mu_1\mu_2} v_{\mu_2} a_{\mu_2\mu_3} \dots v_{\mu_k} a_{\mu_k\mu_{k+1}} v_{\mu_{k+1}}).$$

Then,

$$v_{\mu_{k+1}} \in (F^1)_{\mu_1\mu_2} \cap (F^1)_{\mu_2\mu_3} \cap \dots \cap (F^1)_{\mu_k\mu_{k+1}} \subseteq (F^k)_{\mu_1\mu_{k+1}}.$$

Proof. See proof of lemma 3.2 and decomposition 3.3. □

Theorem 4.3. *In multi digraph g, there is a cycle of length $k \geq 1$ attached to vertex v_i iff $(H^k)_{ii} \neq \emptyset$.*

Proof. For $k = 1$, the theorem is true due to definition 4.1. Let $k > 1$.

Necessity. Let c be a cycle of length k attached to vertex v_i:

$$c = (v_i a_{i\mu_1} v_{\mu_1} a_{\mu_1\mu_2} \dots v_{\mu_{k-1}} a_{\mu_{k-1}i} v_i),$$

- where $a_{xy} \in (S)_{xy}$ are arcs and S is the walk generator. Then the following walk

$$(v_i a_{i\mu_1} v_{\mu_1} a_{\mu_1\mu_2} \dots v_{\mu_{k-1}})$$

is a path of length $k-1$. Then, due to lemma 4.2,

$$v_{\mu_{k-1}} \in (F^{k-1})_{i\mu_{k-1}} \neq \emptyset.$$

On the other hand, due to definition 3.5,

$$v_{\mu_{k-1}} \in (G^1)_{\mu_{k-1}i} \neq \emptyset.$$

Thus, due to definition 4.2,

$$v_{\mu_{k-1}} \in (F^{k-1})_{i\mu_{k-1}} \cap (G^1)_{\mu_{k-1}i} \subseteq (H^k)_{ii} \neq \emptyset.$$

Sufficiency. Let

$$(H^k)_{ii} \neq \emptyset.$$

Then, due to definition 4.2, there exists such couple of sets $(F^{k-1})_{ix}$ and $(G^1)_{xi}$ that

$$(F^{k-1})_{ix} \cap (G^1)_{xi} \neq \emptyset.$$

Then, due to theorem 3.5 and definition 3.5, there exist a path of length $k-1$ from v_i into v_x and an arc from v_x into v_i. These path and arc constitute a cycle of length k. The cycle is attached to vertex v_i. □

Let's estimate computational complexity of theorem 4.3. By definitions 3.5, 4.2 and due to inclusion 3.4,

$$(H^k)_{ij} \subseteq V.$$

Thus, arc H-coloring 4.1 requires $O(n^2)$ operations. And walk H-coloring requires $O(kn^4)$ operations: $O(kn^4)$ operations to calculate F^{k-1}; $O(n^3)$ operations to calculate G^1; and $O(n^2)$ operations at most to calculate each of n diagonal elements of H^k. The total computational complexity is

$$O(kn^4).$$

Particularly, when $k = n$, theorem 4.3 solves the Hamilton cycle problem in time $O(n^5)$. But a simplification is possible. In this case, there is no need for the whole matrix F^{n-1} but only for any one of its strings. That allows solving the Hamiltonian cycle problem with theorem 4.3 in time

$$O(n^4).$$

Obviously, F-coloring and G-coloring may be swapped in theorem 4.3.

Conclusion

The following polynomial time algorithm detects all paths and cycles of all lengths in form of vertex couples (start, finish):

Step 1: Calculate set matrices F^1, G^1, and H^1. Non-empty elements of matrices F^1 and H^1 indicate 1-paths and 1-cycles appropriately;

Steps 2 ÷ n: Calculate set matrices F^k and H^k, $k = 1, 2, \ldots, n$. Non-empty elements of matrices F^k and H^k indicate k-paths and k-cycles appropriately.

The algorithm could be simplified, for example, with vertex aggregation. In the formulas above, the aggregation might be presented with appropriate box set matrices. The modification may be seen as an incorporation of the DJP algorithm's idea.

Further time-simplifications could be achieved by mixing the described "unvisited vertices" walk coloring with a "visited vertices" walk coloring.

Let us emphasize that the algorithm solves a decision problem "exist/not exist". Although, lemma 4.2 or a "visited vertices" walk coloring can be used for restoration/selection of particular paths/cycles.

References

[1] Hamilton, William Rowan. Memorandum respecting a new system of roots of unity. Philosophical Magazine, 12 1856

[2] Hamilton, William Rowan. Account of the Icosian Calculus. Proceedings of the Royal Irish Academy, 6 1858

[3] W.T. Tutte. On Hamiltonian Circuits. J. London Math. Soc. 21, 98-101, 1946.

[4] W.T. Tutte. A Theorem on Planar Graphs. Trans. Amer. Math. Soc. 82(1956), 99-119.

[5] O.A. Ore. A note on Hamiltonian Circuits. Amer. Math. Monthly. 67(1960), 55.

[6] Richard M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, Complexity of Computer Computations, 85-103, New York, 1972 Plenum Press.

[7] Chvtal, Vclav. Tough graphs and Hamiltonian circuits. Discrete Mathematics 5 (3): 215228, 1973

[8] H.A. Jung. On Maximal Circuits in Finite Graphs. Annals of Discrete Math 3(1978), 129-144.

[9] B. Bollabas and A.M. Hobbs. Hamiltonian cycles. Advances in Graph Theory. (B. Bollabas ed) North-Holland Publ., Amsterdam, 1978, 43-48.

[10] M.R. Garey and D.S. Johnson. Computers and Intractability, a Guide to the Theory of NP-Completeness. W.H. Freeman and Co,. San Francisco, 1979.

[11] G.H. Fan. A new Sufficient Condition for cycles in graphs. J. Combinat. Theory B37(1984) 221-227.

[12] Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. In Proc. of the twentieth annual ACM Sympos. on Theory of computing. Chicago, Illinois, 223 - 228, 1988.

[13] D. Bauer, E. Schmeichel, and H.J. Veldman. Some Recent Results on Long Cycles in Tough Graphs. Off Prints from Graph Theory, Combinatorics, and Applications. Ed. Y. Alavi, G. Chartrand, O.R. Ollerman, A.J. Schwenk. John Wiley and Sons, Inc. 1991.

[14] R. Diestel. Graph Theory. New York, Springer, 1997.

[15] The Traveling Salesman Problem and Its Variations. Gregory Gutin and Abraham P. Punnen (Eds.). Kluwer Academic Publishers, 2002.

[16] Bauer, Douglas; Broersma, Hajo; Schmeichel, Edward. Toughness in graphs a survey. Graphs and Combinatorics 22 (1): 135, 2006.

[17] Stephen Cook. Low Level Reverse Mathematics. Plenary Lecture for CiE 2007, 2007.

Genesys Telecommunication Laboratories, Inc.
1255 Treat Blvd., Walnut Creek, CA 94596
E-mail address: sgubin@genesyslab.com